2. Fundamentos de arquitectura multi-tenant

Existen incontables permutaciones de patrones y estrategias de arquitectura multi-tenant que se componen para crear la arquitectura SaaS que mejor se alinea con el dominio, el cumplimiento (compliance) y las realidades del negocio de una empresa SaaS.

Agregar multi-tenancy a tu arquitectura

En las aplicaciones clásicas, el entorno se construye desde cero con la suposición de que será instalado y ejecutado por clientes individuales. Cada cliente esencialmente tiene su propia huella de recursos dedicada.

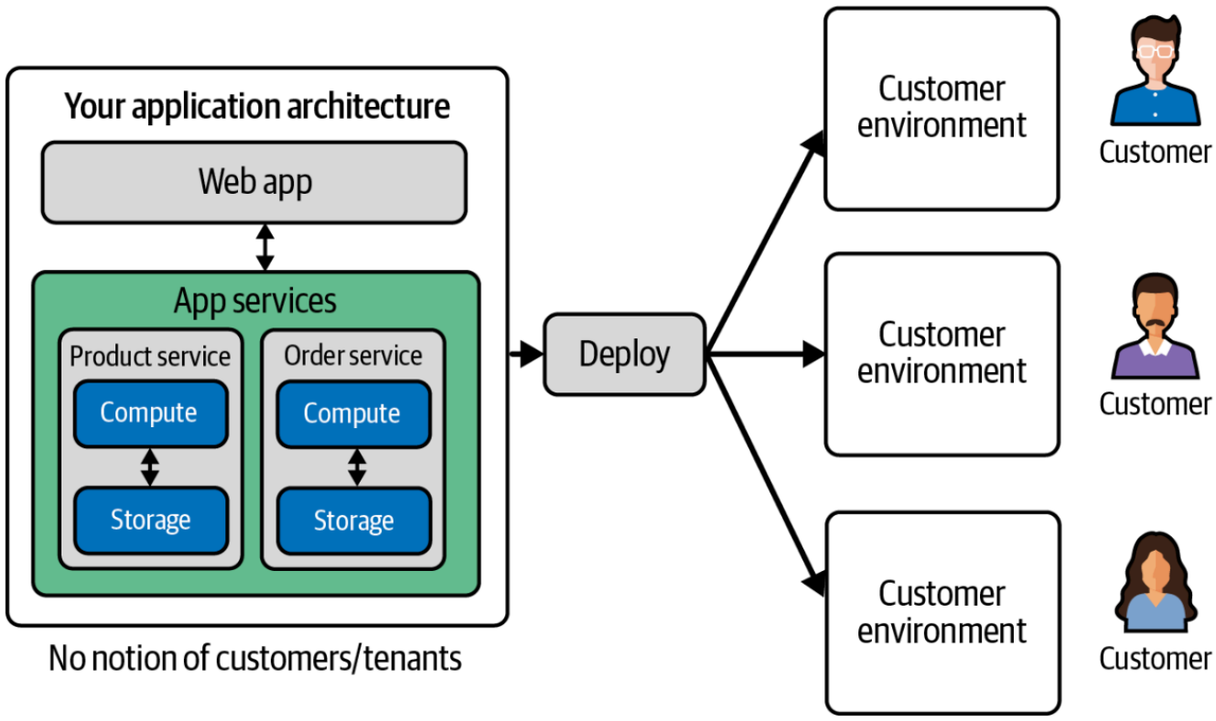

Figure 2-1. Traditional non-SaaS environments

Las decisiones sobre cómo los clientes ingresan al entorno, cómo acceden a nuestros recursos y cómo consumen los servicios de nuestro entorno son mucho más simples cuando sabemos que se ejecutarán en un entorno dedicado a cada cliente.

Ahora, pensemos qué significa entregar esta misma aplicación en un entorno SaaS multi-tenant.

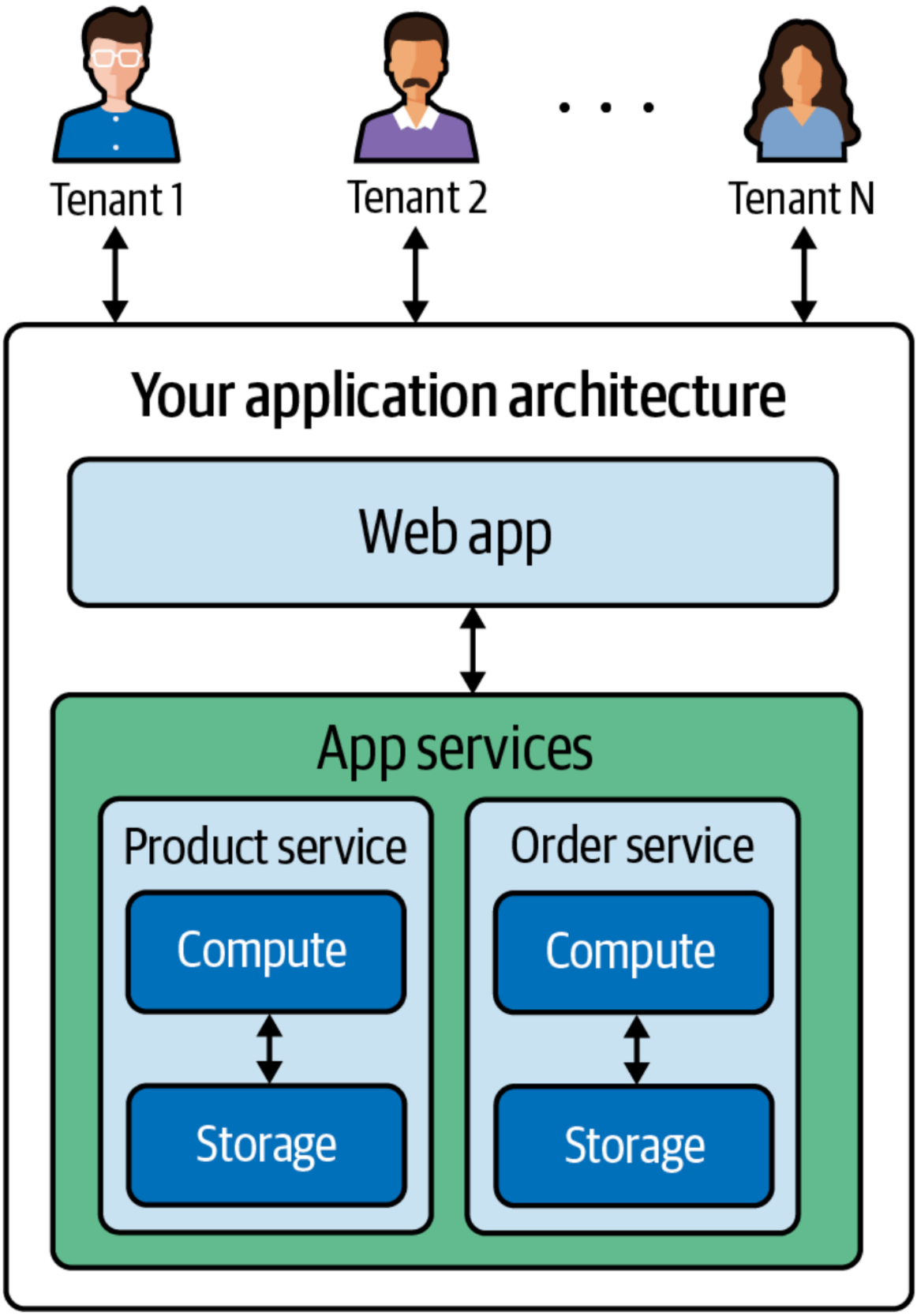

Figure 2-2. The shift to a tenant-centric experience

Dar soporte a este modelo alcanza todas las dimensiones de la implementación subyacente de tu sistema. Afecta cómo implementas la autenticación, routing, escalado, rendimiento, almacenamiento y, en áreas específicas, cómo codificas la lógica de aplicación de tu sistema.

Por ahora, basta con saber que a cada consumidor de nuestro entorno se le denominará tenant. Usaremos esta información de tenant en múltiples capas de nuestras discusiones de arquitectura SaaS. Representa uno de los elementos más fundamentales de cualquier arquitectura SaaS.

Las dos mitades de toda arquitectura SaaS

En el lado derecho del diagrama, verás lo que se etiqueta como el **control plane**. Allí es donde colocaremos todos los componentes transversales, servicios y capacidades que soportan las necesidades fundamentales de un entorno SaaS multi-tenant.

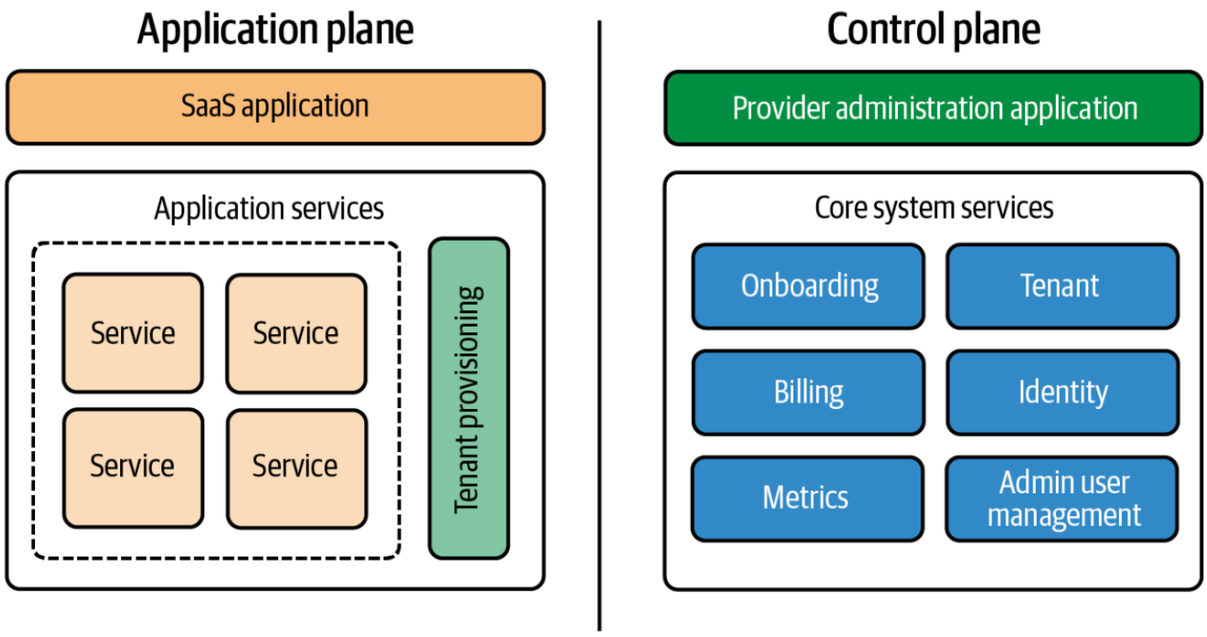

Figure 2-4. SaaS application and control planes

A menudo describimos el control plane como un panel único de control que se usa para orquestar y operar todas las piezas móviles de tu solución SaaS. Él es el núcleo donde se habilitarán muchos de los principios esenciales para el éxito de tu negocio SaaS.

También verás que nuestro control plane incluye una aplicación de administración. Esto representa la consola o experiencia de administración que usa un proveedor SaaS para configurar, gestionar y operar su entorno SaaS.

Aunque los arquitectos y desarrolladores SaaS a menudo se sienten tentados a iniciar la discusión de SaaS con los aspectos multi-tenant de su aplicación, los fundamentos de tu arquitectura SaaS suelen comenzar en el control plane.

En contraste, **el application plane es donde se materializan las funcionalidades y características de tu servicio SaaS.** Aquí vemos la manifestación de los principios multi-tenant que clásicamente se asocian con los entornos SaaS.

Tiendo a ver el application plane como un lienzo en blanco que se pinta según la composición única de servicios y capacidades que requiere mi servicio SaaS.

Dentro del control plane

Onboarding

El control plane es responsable de gestionar y orquestar todos los pasos necesarios para introducir un nuevo tenant en tu entorno SaaS.

Las decisiones que tomes aquí, en muchos aspectos, están en el núcleo de habilitar muchos de los elementos empresariales y de diseño multi-tenant de tu entorno SaaS.

Después de esta solicitud inicial, el control plane posee el resto del flujo de onboarding, creando y configurando nuestro tenant y su correspondiente huella de recursos de identidad. Esto incluye asignar un identificador único a nuestro tenant que se aprovechará en la mayoría de las piezas móviles de nuestra arquitectura multi-tenant.

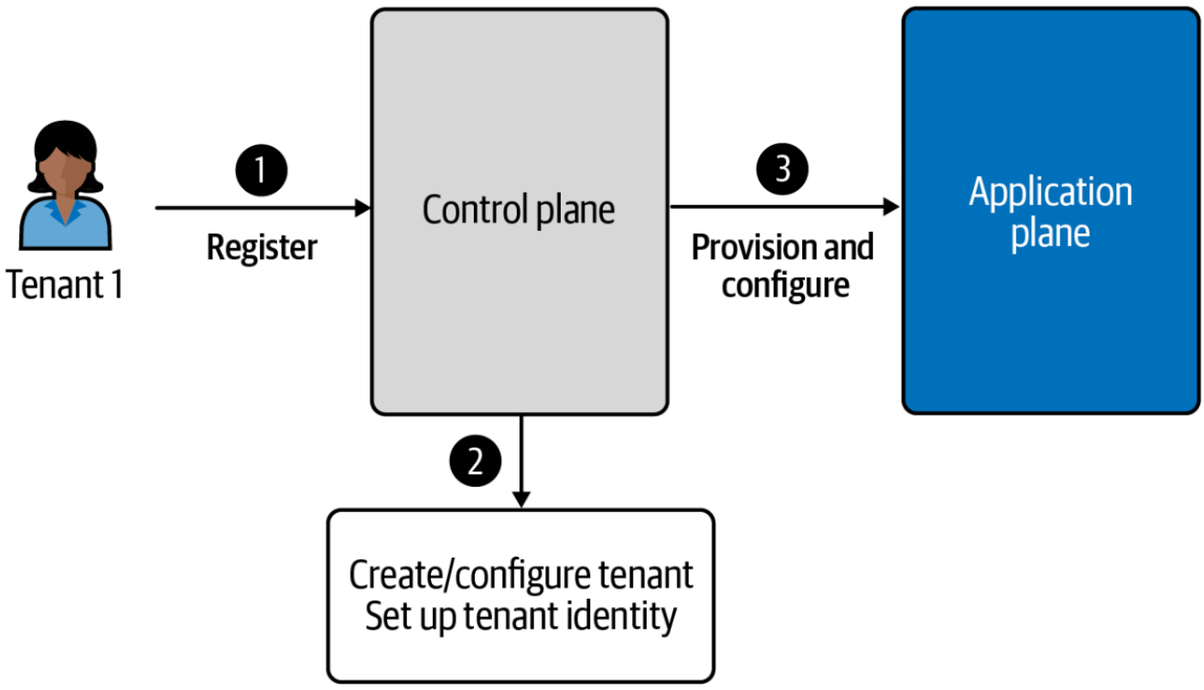

Figure 2-5. Onboarding tenants

También notarás que mostramos el control plane interactuando con el application plane, aprovisionando y configurando cualquier recurso específico de la aplicación que pueda ser necesario para cada tenant.

La conclusión de nivel superior es que onboarding está en el centro de crear y conectar los elementos más básicos de un entorno multi-tenant:

- Tenants,
- Usuarios,
- Identidad,
- y recursos de aplicación de tenant.

Identity

En la izquierda, verás la noción clásica de identidad de usuario que típicamente se asocia con authentication y authorization. Es verdad que nuestro usuario SaaS se autenticará contra nuestro sistema SaaS. Sin embargo, en un entorno multi-tenant, poder autenticar un usuario no es suficiente.

Un sistema SaaS debe saber quién eres como usuario y también debe ser capaz de vincular ese usuario a un tenant. De hecho, cada usuario que está conectado en nuestro sistema debe estar asociado de alguna manera a un tenant. A menudo me refiero a esta vinculación usuario/tenant como tu identidad SaaS.

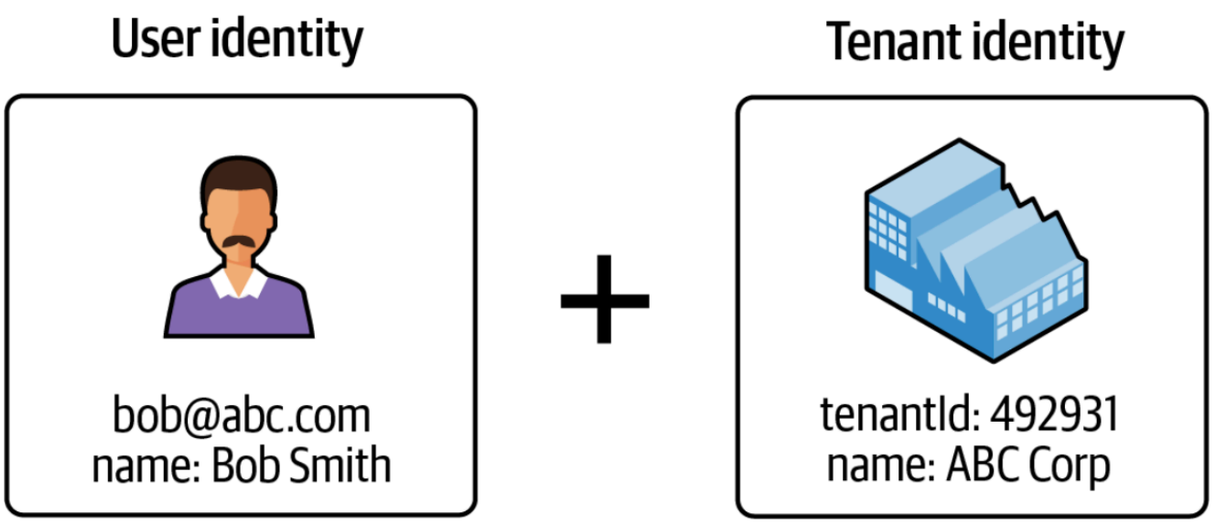

Figure 2-6. Binding users to a tenant identity

Esta noción de mapear usuarios a tenants se ajusta más naturalmente en entornos donde un individuo se vincula a un servicio como parte de una entidad. En un contexto B2B, mi negocio es el tenant y yo soy uno de posiblemente muchos usuarios en ese negocio.

Esta vinculación usuario/tenant termina agregando una complicación a la experiencia general de identidad de nuestro sistema, requiriendo que arquitectos y desarrolladores SaaS desarrollen estrategias para vincular estos dos conceptos de una manera que aún se conforme a los requisitos de tu modelo general de authentication.

Las políticas y patrones que apliques aquí tendrán un impacto en cascada en muchas de las piezas móviles de tu diseño e implementación.

Métricas

Cuando tu aplicación está ejecutándose en un modelo multi-tenant, se vuelve más difícil crear una imagen clara de cómo tus tenants están utilizando tu sistema. Si estás compartiendo infraestructura, por ejemplo, es muy difícil saber qué tenants están consumiendo actualmente esa infraestructura y cómo la actividad de tenants individuales podría estar impactando la escala, rendimiento y disponibilidad de tu solución.

Estos factores hacen que sea especialmente importante para las empresas SaaS invertir en construir una experiencia rica de métricas y análisis como parte de su control plane.

El objetivo es crear un hub centralizado para capturar y agregar la actividad de tenants que permita a los equipos monitorear y analizar el perfil de uso y consumo de tenants individuales.

Los desarrolladores de microservicios necesitarán pensar en cómo y dónde añadirán instrumentación de métricas. Los equipos de infraestructura necesitarán decidir cómo y dónde expondrán la actividad de infraestructura.

> El tenant debe estar en el centro de esta estrategia de métricas. Tener datos sobre consumo y actividad tiene significativamente menos valor si no puede ser filtrado, analizado y visualizado a través de la lente de tenants individuales.

Billing

La facturación tiene un par de puntos de contacto dentro del control plane. Por lo general, está conectado a la experiencia de onboarding, donde cada nuevo tenant debe ser creado como un “customer” dentro de tu sistema de facturación. Esto puede incluir configurar el plan de facturación del tenant y establecer otros atributos del perfil de facturación del tenant.

Muchas soluciones SaaS tienen estrategias de facturación que miden y cuantifican la actividad de los tenants como parte de la generación de una factura.

Tenant management

Cada tenant en nuestro sistema SaaS necesita ser centralmente gestionado y configurado.

Típicamente, este es un servicio bastante básico que proporciona todas las operaciones necesarias para crear y gestionar el estado de los tenants. Esto incluye rastrear atributos clave que asocian tenants con un identificador único, planes de facturación, políticas de seguridad, configuración de identidad y un estado activo/inactivo.

Esto proporciona un único punto de configuración de tenant que permite que los tenants sean fácilmente gestionados a través de una experiencia centralizada.

Dentro del application plane

A medida que profundices en el application plane, descubrirás que tu stack de tecnología y huella de despliegue tendrán una influencia significativa en cómo se aplican estos conceptos.

Contexto de tenant

El tenant context no se asigna a ninguna estrategia o mecanismo específico. En su lugar, es un concepto más amplio que pretende transmitir la idea de que nuestro application plane siempre funciona en el contexto de tenants específicos.

Este contexto a menudo se representa como un token u otra construcción que empaqueta todos los atributos de tu tenant.

Ahora, verás que este tenant context tiene una influencia directa en cómo tu arquitectura de aplicación procesa las solicitudes de tenants. Esto puede afectar:

- Routing,
- Logging,
- Métricas,
- Acceso a datos,
- y una serie de otras construcciones que viven dentro del application plane.

Esto comienza en el lado izquierdo del diagrama donde los tenants se autentican contra la identidad que se creó durante el onboarding y adquieren su tenant context. Este contexto se inyecta entonces en un servicio de la aplicación. Este mismo contexto fluye hacia cada interacción posterior del sistema, permitiéndote adquirir y aplicar ese contexto en una serie de casos de uso diferentes.

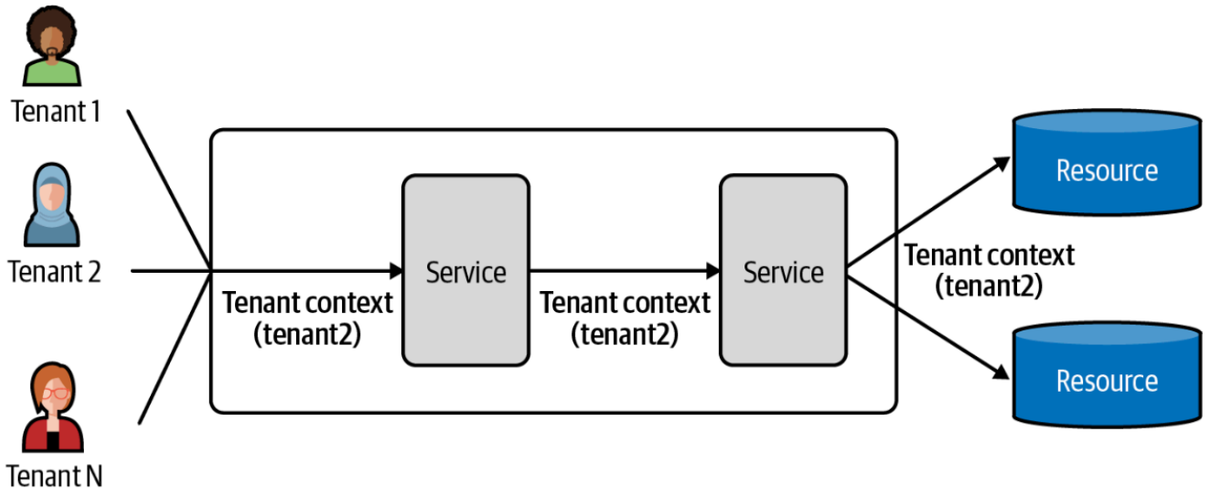

Figure 2-7. Applying tenant context

Esto representa una de las diferencias más fundamentales de un entorno SaaS. Nuestros servicios no solo funcionan con usuarios, ellos deben incorporar el tenant context como parte de la implementación de todas las piezas móviles de nuestra aplicación SaaS.

Como arquitecto de SaaS, esto significa que debes considerar de forma permanente cómo se propagará el contexto del tenant a través del sistema. También tendrás que estar pensando en las estrategias de tecnología específica que se utilizarán para empaquetar y aplicar este *tenant context* de maneras que limiten la complejidad y promuevan la agilidad. Este es un acto de equilibrio continuo para los arquitectos y desarrolladores SaaS.

Aislamiento de tenant

El multi-tenancy, por su propia naturaleza, se enfoca precisamente en colocar a nuestros clientes y sus recursos en entornos donde los recursos pueden ser compartidos o al menos estar uno al lado del otro en entornos de infraestructura común.

Esta realidad significa que las soluciones multi-tenant a menudo deben aplicar e implementar medidas creativas para asegurar que los recursos del tenant estén protegidos contra cualquier potencial acceso entre tenants (cross-tenant access).

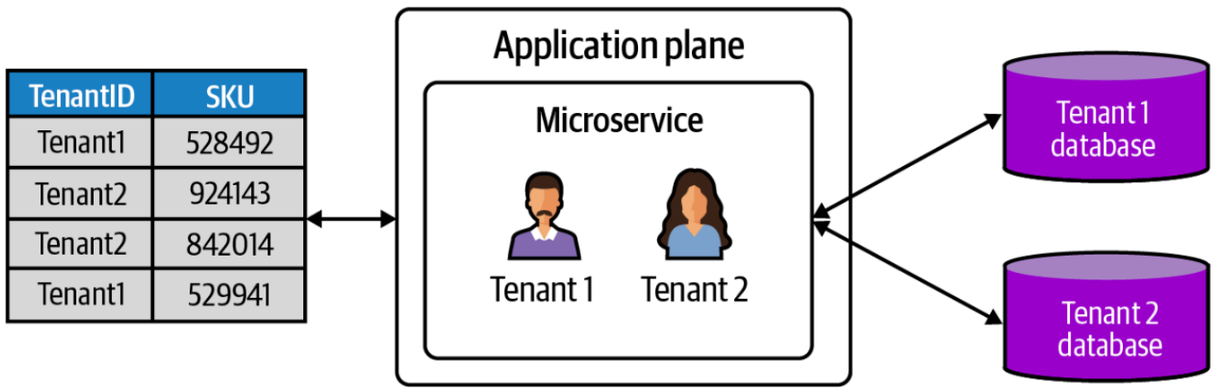

Figure 2-8. Implementing tenant isolation

Si bien los datos para nuestros tenants se almacenan en construcciones de almacenamiento separadas, **no hay nada en nuestra solución que garantice que el Tenant 1 no pueda acceder a los datos del Tenant 2.** Esto es cierto incluso si mis bases de datos son separadas. El hecho de poner los datos en una base de datos separada no garantiza que los tenants no puedan cruzar esta frontera.

Para prevenir cualquier acceso a los recursos de otro tenant, nuestro application plane debe introducir una construcción que prevenga este acceso entre tenants (cross-tenant access).

Particionamiento de datos

Los servicios y capacidades dentro de nuestro *application plane* a menudo necesitan almacenar datos para los tenants. Por supuesto, cómo y dónde elijas almacenar esos datos puede variar significativamente según el perfil multi-tenant de tu aplicación SaaS.

En el mundo del almacenamiento multi-tenant, nos referimos al diseño de estos diferentes modelos de almacenamiento como data partitioning. **La idea clave es que estás seleccionando una estrategia de almacenamiento que divide los datos del tenant según el perfil multi-tenant de esos datos.**

Como arquitecto de SaaS, tu trabajo será examinar el rango de diferentes datos almacenados por tu sistema y determinar qué estrategia de particionamiento se alinea mejor con tus necesidades.

También es importante notar que, cuando eliges una estrategia, esta es a menudo una decisión granular. Cómo particionar los datos puede variar entre los diferentes servicios dentro de tu application plane.

Enrutamiento de tenant

En el modelo de arquitectura SaaS más simple, es posible que todos los tenants compartan sus recursos. Sin embargo, en la mayoría de los casos, tu arquitectura va a tener variaciones donde algunos o todos los recursos del tenant pueden ser dedicados.

El punto principal es que las arquitecturas de aplicación SaaS a menudo deben soportar una huella de despliegue distribuida que tenga cualquier número de recursos ejecutándose en una combinación de modelos compartidos y dedicados.

En este ejemplo, tenemos tres tenants que estarán haciendo solicitudes para invocar operaciones en nuestros servicios de aplicación. En este ejemplo particular, tenemos algunos recursos que son compartidos y algunos que son dedicados.

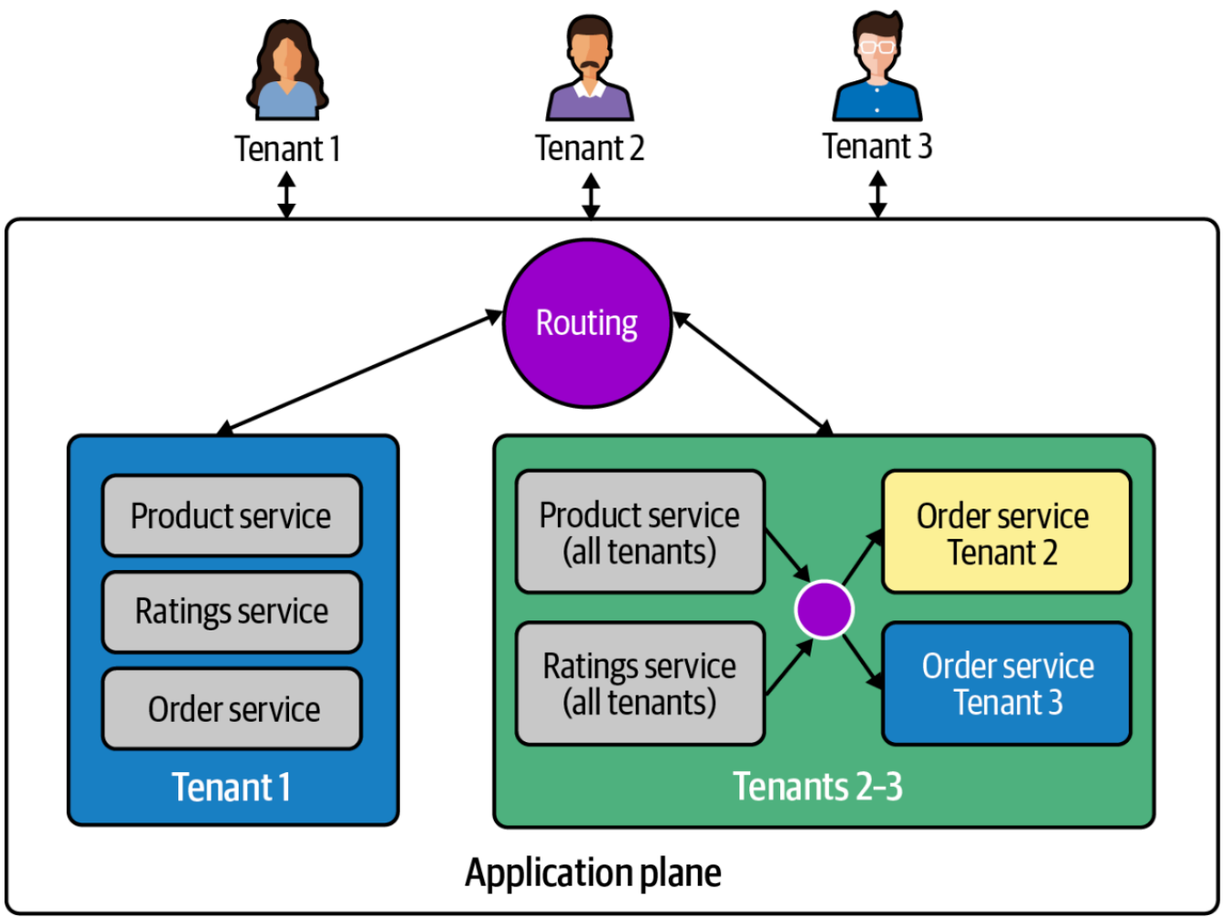

Figure 2-9. Routing on tenant context

Ahora, si observas la configuración general de estos servicios, puedes ver dónde nuestra arquitectura multi-tenant tendría que incluir estrategias y construcciones que enruten correctamente las solicitudes de los tenants a los servicios apropiados.

Este *router* debe aceptar solicitudes de todos los tenants y usar el tenant context inyectado (que discutimos anteriormente) para determinar cómo y dónde enrutar cada solicitud.

Este ejemplo está destinado a destacar la necesidad de construcciones de routing conscientes de multi-tenant que puedan manejar los diversos despliegues que podríamos tener en un entorno SaaS.

Despliegue de aplicación multi-tenant

Ya hemos notado que los tenants pueden tener una mezcla de recursos dedicados y compartidos. Algunos pueden tener recursos totalmente dedicados, algunos pueden tener totalmente compartidos, y otros pueden tener alguna mezcla de dedicados y compartidos. Sabiendo esto, **ahora tenemos que considerar cómo esto influenciará la implementación de DevOps de nuestro despliegue de aplicación.**

La zona gris

Aunque el *control plane* y el *application plane* cubren la mayoría de los constructos de arquitectura multi-tenant fundamentales, aún hay algunos conceptos que no se ajustan claramente en ninguno de ellos.

Tiers (Niveles)

Tiering es una estrategia que la mayoría de los arquitectos han encontrado como parte del consumo de varias ofertas de terceros. La idea básica aquí es que las empresas SaaS usan niveles (tiers) para crear diferentes variaciones de una oferta con puntos de precio separados.

El error que algunos arquitectos y desarrolladores SaaS cometen es asumir que estos *tiers* son principalmente estrategias de precios y empaquetamiento. En realidad, tiering puede tener un impacto significativo en muchas de las dimensiones de tu

arquitectura multi-tenant.

Tiering se superpone naturalmente a nuestra discusión del tenant context, ya que el contexto que se comparte entre nuestra arquitectura a menudo incluye una referencia al tier del tenant dado.

En algunas implementaciones de tiering, veremos que los equipos colocan esto dentro de su control plane como un concepto de primera clase.

Los tiers también a menudo se correlacionan con un plan de facturación, lo que parecería natural mantener dentro del alcance del control plane.

Al mismo tiempo, los tiers también se utilizan mucho dentro del *application plane*. Se pueden usar para configurar estrategias de routing o también podrían ser referenciados como parte de la configuración de políticas de throttling.

Usuarios: Tenant, Administrador de tenant y Administrador del sistema

En un entorno multi-tenant, tenemos múltiples nociones de lo que significa ser un usuario, cada una de las cuales juega un rol distinto.

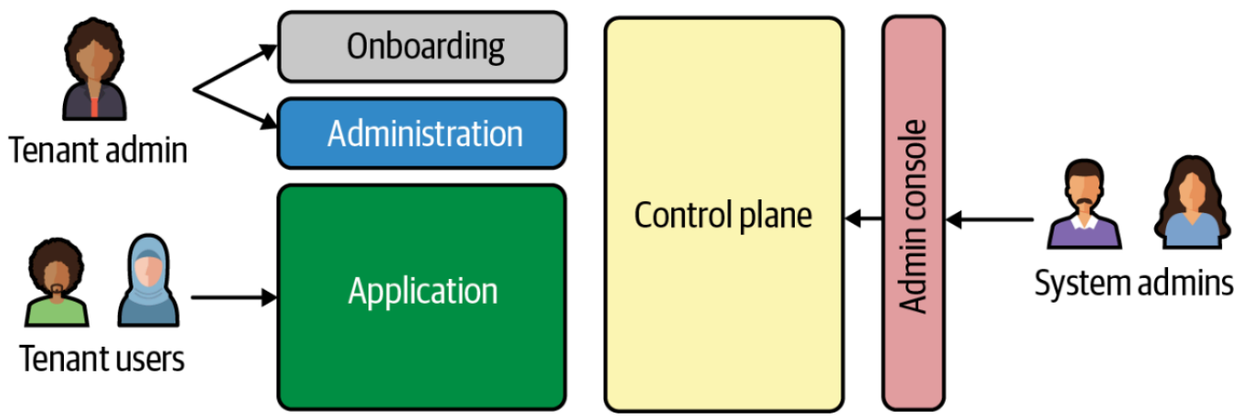

Figure 2-10. Multi-tenant user roles

En el lado izquierdo del diagrama, verás que tenemos los roles típicos relacionados con tenants. Hay dos tipos distintos de roles aquí: administradores de tenant y usuarios de tenant.

Un administrador de tenant representa el usuario inicial de tu tenant que es incorporado al sistema. Este usuario generalmente recibe privilegios de administrador. Esto les permite acceder a la funcionalidad única de administración de aplicación que se utiliza para configurar, gestionar y mantener constructos a nivel de aplicación.

En el lado derecho del diagrama, verás que también tenemos administradores del sistema. Estos usuarios están conectados al proveedor SaaS y tienen acceso al control plane de tu entorno para gestionar, operar y analizar la salud y actividad de un entorno SaaS.

Notarás que también he mostrado una consola de administración como parte del control plane. Esto representa a menudo una parte pasada por alto del rol de administrador del sistema. Está aquí para destacar la necesidad de una consola de administración SaaS específica que se utiliza para gestionar, configurar y operar tus tenants.

Los arquitectos SaaS necesitan considerar cada uno de estos roles cuando construyen un entorno multi-tenant.

Aprovisionamiento de tenant

También observamos cómo el proceso de onboarding puede necesitar aprovisionar y configurar la infraestructura de aplicación como parte de la experiencia de onboarding.

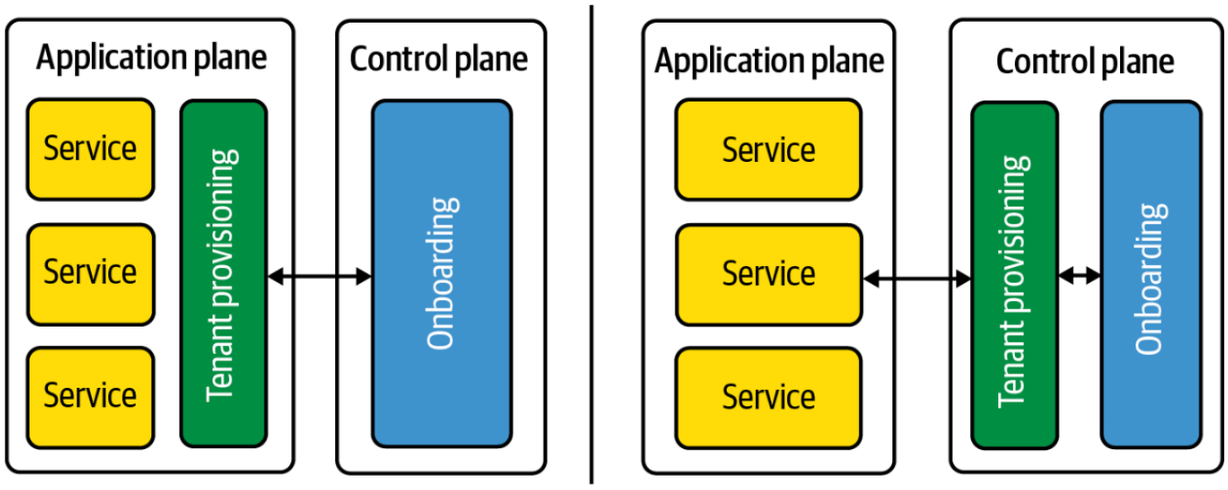

Figure 2-11. Placing the tenant provisioning process

Los trade-offs se centran en el encapsulamiento y abstracción del application plane. Si crees que la estructura y la huella de la infraestructura de aplicación deben ser desconocidas para el control plane, entonces favorecerás el modelo de la izquierda. Si sientes que el onboarding ya es propiedad del control plane, podrías argumentar que es natural que también sea dueño del proceso de aprovisionamiento y configuración de la aplicación.

Integración del Control Plane y Application Plane

Algunas organizaciones crearán límites muy específicos entre el *control plane* y el a*pplication plane*. Esto podría ser un network boundary o algún otro constructo arquitectónico que separe estos dos planos.

Algunos equipos pueden optar por un modelo más débilmente acoplado que sea más impulsado por eventos o mensajes, mientras que otros pueden requerir una integración más nativa que permita un control más directo de los recursos del application plane.

La clave es reflexionar al elegir un modelo de integración que habilite el nivel de control que se ajuste a las necesidades de tu dominio, aplicación y entorno particular.

Selección de tecnologías para el Control Plane y Application Plane

Los equipos SaaS eligen las tecnologías para implementar sus soluciones SaaS basándose en cualquier número de variables diferentes. Conjuntos de habilidades, proveedores de nube, necesidades del dominio, consideraciones de legacy.

Las decisiones también pueden ser mucho más granulares. Podría elegir una tecnología para algunos tipos de servicios y diferentes tecnologías para otros servicios. La clave es que no debes asumir que el perfil, consumo y perfil de rendimiento de tu control plane y application plane serán iguales. **Como parte de la arquitectura de tu entorno SaaS, quieres considerar las necesidades tecnológicas de estos dos planos de manera independiente.**